Scaling Distributed Machine Learning with the Parameter Server

By M. Li, D. G. Andersen, J. W. Park, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, B. Su, and A. J. Smola, OSDI 2014.

Presented by Marquita Ellis, CS294 Scalable Machine Learning, February 2016

The World We Live in Now

Real world training data sizes

1TB - 1PB

resulting in models with

10^9 - 10^{12} parameters.

The necessity of fault tolerance at scale

≈ #machine × time	# of jobs	failure rate
100 hours	13,187	7.8%
1, 000 hours	1,366	13.7%
10, 000 hours	77	24.7%

Table 1: Statistics of machine learning jobs for a three month period in a data center.

Easily accessible clouds

Goal

A distributed system supporting efficient execution of Machine Learning algorithms used daily on “trillions of trillion-length feature vectors” possibly even in real-time.

Parameter Server Architecture

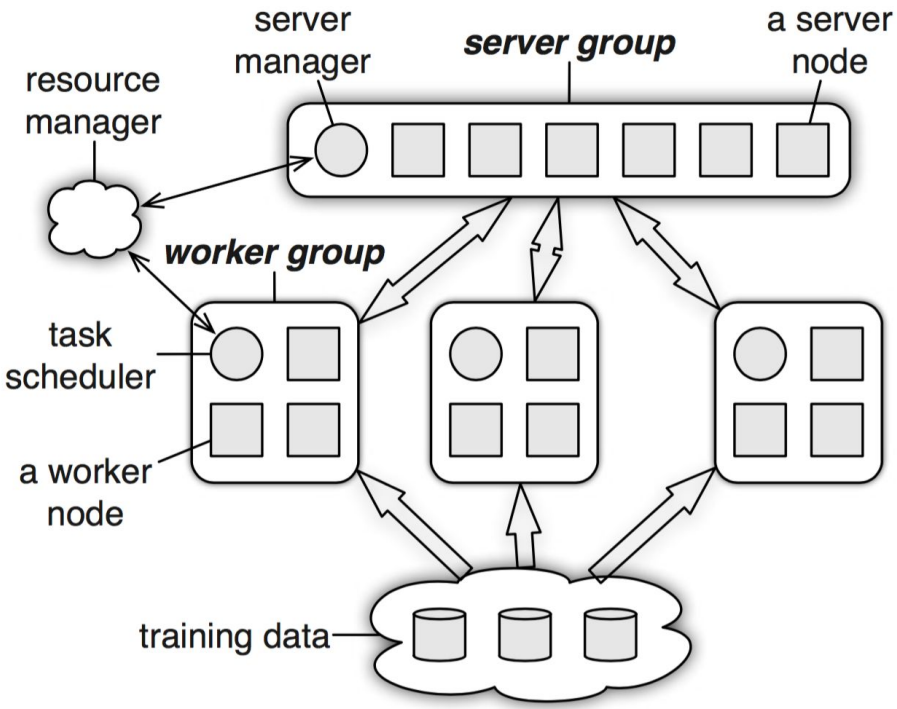


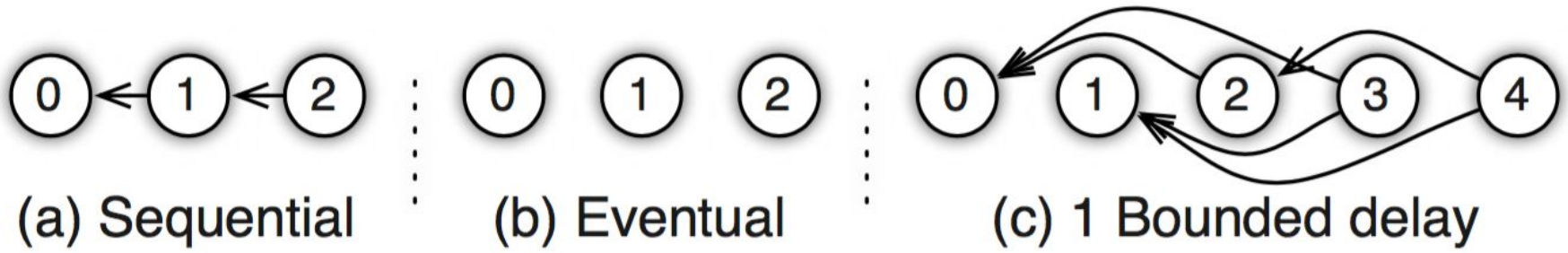
Figure 4: Architecture of a parameter server communicating with several groups of workers.

System Aspects

- Server and worker elasticity (4.5-4.6)
- (Key, Value) vector data representation (3.1)
  - with ordering for vector and matrix semantics
- Asynchronous tasks with optional user-defined dependencies (3.4)
- Range-based communication batching for efficient communication (3.2, 4.2)
- Range-based vector clocks supporting fast recovery and consistency (4.1)
- Fault tolerance via replication optimized with aggregation (4.4)
- Flexible consistency (3.5)

Flexible Consistency

Key insight: there's a trade-off between system efficiency and algorithm convergence rate



Tasks can start only after previous ones have finished

t = 0

All tasks can run simultaneously. The system becomes consistent *eventually.*

t = infinity

A maximal delay time (or number of iterations) *t* is set such that, any new task blocks until all the tasks *t* timesteps ago have completed.

Sparse Logistic Regression Results

- System A and System B are special purpose systems “developed by a large [unnamed] internet company”.
- A and B are > 10k lines of code
- Parameter Server requires 300 lines of code for the same functionality as B

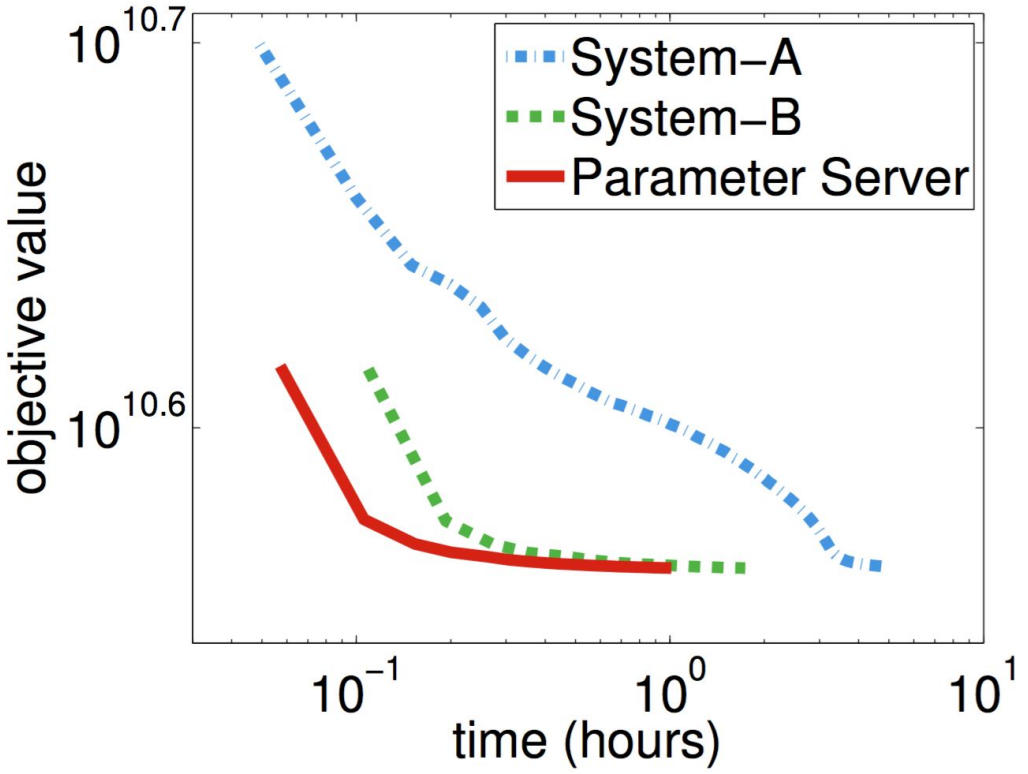


Figure 9: Convergence of sparse logistic regression. The goal is to minimize the objective rapidly.

Sparse Logistic Regression Results Continued

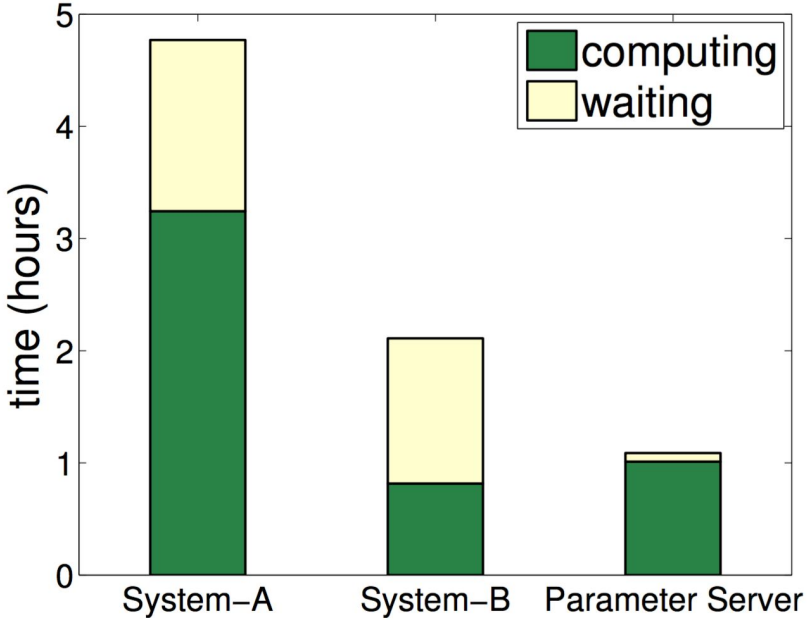


Figure 10: Time per worker spent on computation and waiting during sparse logistic regression.

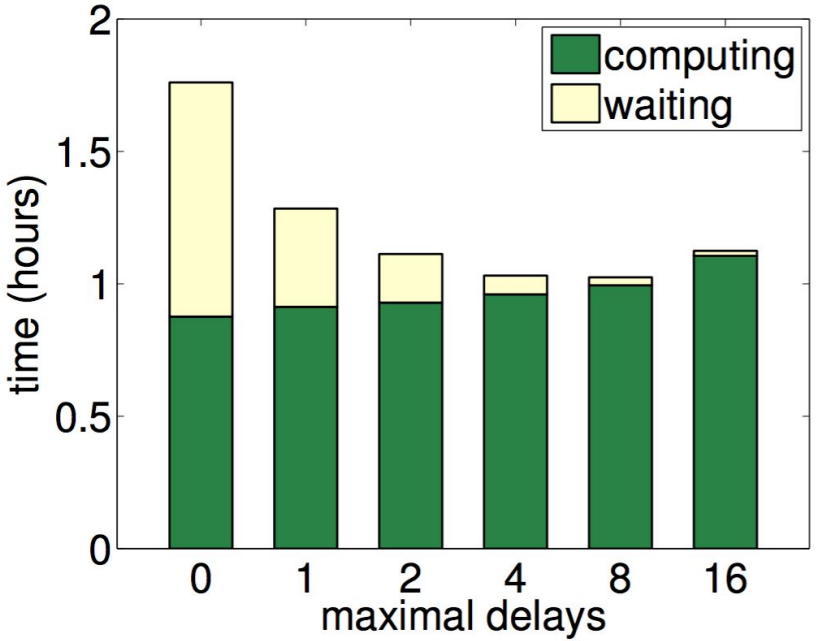


Figure 13: Time a worker spent to achieve the same convergence criteria by different maximal delays.

Other Results

- Latent Dirichlet Allocation (LDA)
  - (nothing surprising)
  - handles more data than any other LDA system the authors are aware of
  - they show scaling speeds up convergence
  - and the importance of coping with stragglers and fault tolerance
- Sketches for generality evaluation
  - performs bulk communication with compressed messages
  - peak and average operations per second and time to recover failed node measured
  - impressive numbers (other systems for comparison not included)

Peak inserts per second	1.3 billion
Average inserts per second	1.1 billion
Peak net bandwidth per machine	4.37 GBit/s
Time to recover a failed node	0.8 second

Table 5: Results of distributed CountMin

Goal

A distributed system supporting efficient execution of Machine Learning algorithms used daily on “trillions of trillion-length feature vectors” possibly even in real-time.

Resulting Design Summary

- asynchronous **communication** with batching, exploiting ML algorithm design
- fast (< 1s) non-disruptive recovery **fault tolerance**
  - with vector clocks ensuring well-defined behavior
- various levels of **consistency** from which the algorithm designer can choose
- vector and matrix global parameter representation to support **ease-of-use** for ML application development
- non-restarting elastic **scalability**
- **generality** across “the most popular” machine learning algorithms